
pysimavr Documentation

Release 0.0.4

ponty

November 10, 2011

CONTENTS

1	Basic usage	2
2	Installation	3
2.1	General	3
2.2	Ubuntu	3
2.3	Uninstall	4
3	Usage	5
3.1	vcd export example	6
3.2	unit test example	7
4	File hierarchy	9
5	update simavr sources	10
6	API	11
6.1	low level interface	11
6.2	high level interface	17
7	Development	20
7.1	Tools	20
7.2	Install on ubuntu	20
7.3	Tasks	21
8	Indices and tables	22
	Python Module Index	23
	Index	24

pysimavr

Date November 10, 2011

PDF [pysimavr.pdf](#)

Contents:

pysimavr is a python wrapper for [simavr](#) which is [AVR](#) and [arduino](#) simulator

Links:

- home: <https://github.com/ponty/pysimavr>
- documentation: <http://ponty.github.com/pysimavr>

Features:

- python wrapper using [swig](#)
- [simavr](#) source code is included for easier installation
- object oriented interface on top of the generated interface
- maximum speed can be real-time
- serial communication
- check [simavr](#) documentation

Known problems:

- included [simavr](#) source code is not up to date
- Python 3 is not supported
- tested only on linux
- more tests needed
- PWM simulation is not real-time
- missing PWM modes
- a lot of messages on stdout
- LCD simulator is not fully implemented

Possible usage:

- unit test
- simulator

Similar projects:

- [simavr](#)
- [emulino](#)
- [Arduino Unit](#)
- [arduemu](#)

BASIC USAGE

```
>>> from pysimavr.avr import Avr
>>> avr=Avr(mcu='atmega48',f_cpu=8000000)
>>> firmware = Firmware('lcd.elf')
>>> avr.load_firmware(firmware)

>>> from pysimavr.sim import ArduinoSim
>>> print ArduinoSim(snippet='Serial.print("hello!");').get_serial()
hello!
```

INSTALLATION

check `simavr` doc: <http://gitorious.org/simavr/pages/GetStarted>

ignore these in `simavr` doc:

- OpenGL (freeglut)
- gcc-avr
- avr-libc
- make

2.1 General

- install `python`
- install `setuptools`
- install `swig` (for source build only)
- install header files and a static library for Python (for source build only)
- install a compiler (for source build only)
- install elf library
- install the program:

```
# as root
easy_install pysimavr
```

2.2 Ubuntu

```
sudo apt-get install python-setuptools
sudo apt-get install swig
sudo apt-get install python-dev
sudo apt-get install gcc
sudo apt-get install libelf-dev
sudo easy_install pysimavr
```

2.3 Uninstall

first install `pip`:

```
# as root
pip uninstall pysimavr
```

USAGE

pysimavr.examples.simple:

```
from pysimavr.avr import Avr

avr=Avr(mcu='atmega48',f_cpu=8000000)
avr.step(1)
print avr.pc

$ python -m pysimavr.examples.simple
Starting atmega48 - flashend 0fff ramend 02ff e2end 00ff
atmega48 init
2
```

pysimavr.examples.hello:

```
from pysimavr.sim import ArduinoSim
from entrypoint2 import entrypoint

@entrypoint
def run_sim():
    print ArduinoSim(snippet='Serial.print("hello!");').get_serial()

$ python -m pysimavr.examples.hello
Loaded 2148 .text
Loaded 26 .data
Starting atmega328 - flashend 7fff ramend 08ff e2end 03ff
atmega328 init
uart_udp_init bridge on port 4321
avr_timer_reconfigure-0 clock turned off
avr_timer_reconfigure-0 clock turned off
avr_timer_configure-0 TOP 7812.50Hz = 2048 cycles
avr_timer_configure-0 C 8888.89Hz = 1800 cycles
avr_timer_configure-0 TOP 976.56Hz = 16384 cycles
avr_timer_configure-0 C 1111.11Hz = 14400 cycles
avr_timer_configure-1 TOP 30.52Hz = 524288 cycles
avr_timer_configure-1 C 8888.89Hz = 1800 cycles
avr_timer_configure-1 TOP 3.81Hz = 4194304 cycles
avr_timer_configure-1 C 1111.11Hz = 14400 cycles
avr_timer_reconfigure-1 unsupported timer mode wgm=1 (0)
avr_timer_configure-2 TOP 976.56Hz = 16384 cycles
avr_timer_configure-2 C 1111.11Hz = 14400 cycles
avr_timer_reconfigure-2 unsupported timer mode wgm=1 (0)
ADC Start AREF 0 AVCC 5000
UART-0 configured to 00cf = 4807 bps, 5 data 1 stop
```


Roughly 1666 usec per bytes
hello!

3.1 vcd export example

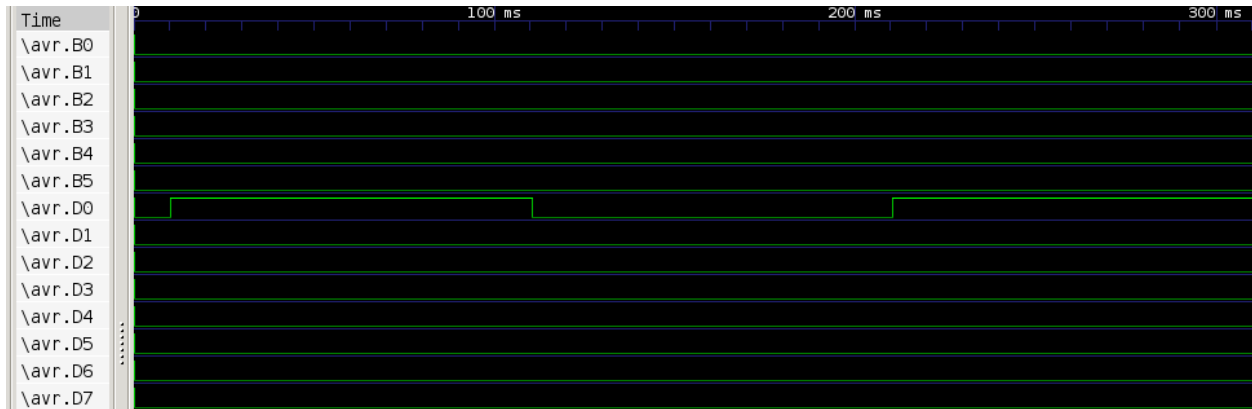
pysimavr.examples.vcd:

```
from entrypoint2 import entrypoint
from pysimavr.sim import ArduinoSim

@entrypoint
def run_sim(vcdfile='delay.vcd'):
    snippet='''
        Serial.println("start");
        pinMode(0, OUTPUT);
        digitalWrite(0, HIGH);
        delay(100);
        digitalWrite(0, LOW);
        delay(100);
        digitalWrite(0, HIGH);
        delay(100);
        digitalWrite(0, LOW);
        delay(100);
        Serial.println("end");
    '''

    sim=ArduinoSim(snippet=snippet, vcd=vcdfile, timespan=0.5)
    sim.run()
    print sim.serial

>>> from pysimavr.examples.vcd import run_sim
>>> run_sim(vcdfile='docs/vcd.vcd')
Loaded 2954 .text
Loaded 30 .data
Starting atmega328 - flashend 7fff ramend 08ff e2end 03ff
atmega328 init
uart_udp_init bridge on port 4321
avr_timer_reconfigure-0 clock turned off
avr_timer_reconfigure-0 clock turned off
avr_timer_configure-0 TOP 7812.50Hz = 2048 cycles
avr_timer_configure-0 C 15151.52Hz = 1055 cycles
avr_timer_configure-0 TOP 976.56Hz = 16384 cycles
avr_timer_configure-0 C 1893.94Hz = 8447 cycles
avr_timer_configure-1 TOP 30.52Hz = 524288 cycles
avr_timer_configure-1 C 15151.52Hz = 1055 cycles
avr_timer_configure-1 TOP 3.81Hz = 4194304 cycles
avr_timer_configure-1 C 1893.94Hz = 8447 cycles
avr_timer_reconfigure-1 unsupported timer mode wgm=1 (0)
avr_timer_configure-2 TOP 976.56Hz = 16384 cycles
avr_timer_configure-2 C 1893.94Hz = 8447 cycles
avr_timer_reconfigure-2 unsupported timer mode wgm=1 (0)
ADC Start AREF 0 AVCC 5000
UART-0 configured to 00cf = 4807 bps, 5 data 1 stop
Roughly 1666 usec per bytes
start
end
```



3.2 unit test example

pysimavr/examples/test_example.py

```
''' unit test example'''
```

```
from pysimavr.sim import ArduinoSim
```

```
def test_atmega88():
    mcu = 'atmega88'
    snippet = 'Serial.print("hi");'

    output = ArduinoSim(snippet=snippet, mcu=mcu, timespan=0.01).get_serial()
    assert output == 'hi'
```

```
$ nosetests --verbose pysimavr/examples/test_example.py
pysimavr.examples.test_example.test_atmega88 ... ok
```

```
-----
Ran 1 test in 2.318s
```

OK

Loaded 2068 .text

Loaded 22 .data

Starting atmega88 - flashend 1fff ramend 04ff e2end 01ff

atmega88 init

uart_udp_init bridge on port 4321

avr_timer_reconfigure-0 clock turned off

avr_timer_reconfigure-0 clock turned off

avr_timer_configure-0 TOP 7812.50Hz = 2048 cycles

avr_timer_configure-0 C 10638.30Hz = 1504 cycles

avr_timer_configure-0 TOP 976.56Hz = 16384 cycles

avr_timer_configure-0 C 1329.79Hz = 12032 cycles

avr_timer_configure-1 TOP 30.52Hz = 524288 cycles

avr_timer_configure-1 C 10638.30Hz = 1504 cycles

avr_timer_configure-1 TOP 3.81Hz = 4194304 cycles

avr_timer_configure-1 C 1329.79Hz = 12032 cycles

avr_timer_reconfigure-1 unsupported timer mode wgm=1 (0)

avr_timer_configure-2 TOP 976.56Hz = 16384 cycles

avr_timer_configure-2 C 1329.79Hz = 12032 cycles

avr_timer_reconfigure-2 unsupported timer mode wgm=1 (0)

ADC Start AREF 0 AVCC 5000

UART-0 configured to 00cf = 4807 bps, 5 data 1 stop
Roughly 1666 usec per bytes

FILE HIERARCHY

-docs	sphinx documentation
---_build	generated documentation
-pysimavr	main python package, high level classes
---examples	examples
---swig	all swig files (simavr and parts)
-----cores	copy from simavr
-----include	copy from simavr
-----avr	copy from avr-libc
-----parts	some electronic parts in c
-----sim	copy from simavr
-tests	unit tests

UPDATE SIMAVR SOURCES

copy:

```
$SIMAVR/include      ->  $PYSIMAVR/pysimavr/swig/include
$SIMAVR/simavr/cores ->  $PYSIMAVR/pysimavr/swig/cores
$SIMAVR/simavr/sim   ->  $PYSIMAVR/pysimavr/swig/sim

$AVR_LIBC_INCLUDE/avr ->  $PYSIMAVR/pysimavr/swig/include/avr

(Ubuntu: /usr/lib/avr/include/avr/)
```

API

There are 2 interfaces:

- `pysimavr.swig.*`: low level, generated by swig
- `pysimavr.*`: high level classes, they can redirect function calls to low level interface. Example: `Avr` class (high level) has all properties and methods of `avr_t` class (low level) automatically.

6.1 low level interface

```
class pysimavr.swig.ac_input.ac_input_t
```

```
    avr
    irq
    value
```

```
class pysimavr.swig.hd44780.hd44780_t
```

```
    avr
    cursor
    datapins
    flags
    h
    irq
    pinstate
    readpins
    vram
    w
```

```
class pysimavr.swig.inverter.inverter_t
```

```
    avr
    irq
```

out

class pysimavr.swig.ledrow.ledrow_t

avr

irq

pinstate

pinstate_changed

class pysimavr.swig.sgm7.sgm7_t

avr

digit_count

digit_pin

digit_port

digit_segments

digit_segments_changed

irq

pinstate

segment_pin

segment_port

class pysimavr.swig.simavr.avr_io_t

avr

dealloc

ioctl

irq

irq_count

irq_ioctl_get

irq_names

kind

next

reset

class pysimavr.swig.simavr.avr_iopin_t

pin

port

class pysimavr.swig.simavr.avr_ioport_getirq_t

bit

```
    irq
class pysimavr.swig.simavr.avr_ioport_state_t

    ddr
    name
    pin
    port
class pysimavr.swig.simavr.avr_ioport_t

    io
    name
    pcint
    r_ddr
    r_pcint
    r_pin
    r_port
class pysimavr.swig.simavr.avr_irq_pool_t

    count
    irq
class pysimavr.swig.simavr.avr_irq_t

    flags
    hook
    irq
    name
    pool
    value
class pysimavr.swig.simavr.avr_kind_t

    make
    names
class pysimavr.swig.simavr.avr_symbol_t

    addr
    symbol
class pysimavr.swig.simavr.avr_t
```


aref
avcc
codeend
cycle
cycle_timer
cycle_timer_map
data
e2end
eind
flash
flashend
frequency
fuse
gdb
gdb_port
i_shadow
init
io
io_port
io_shared_io
io_shared_io_count
irq_pool
log
mmcu
next_cycle_timer
pc
pending
pending_wait
ramend
rampz
reset
run
signature
sleep
special_deinit
special_init

sreg
state
trace
trace_data
vcc
vcd
vector
vector_size

class pysimavr.swig.simavr.**avr_t_cycle_timer**

param
timer
when

class pysimavr.swig.simavr.**avr_t_io**

irq
r
w

class pysimavr.swig.simavr.**avr_t_io_r**

c
param

class pysimavr.swig.simavr.**avr_t_io_shared_io**

io
used

class pysimavr.swig.simavr.**avr_t_io_shared_io_io**

c
param

class pysimavr.swig.simavr.**avr_t_io_w**

c
param

class pysimavr.swig.simavr.**avr_trace_data_t**

codeline
old

```
    old_pci
    touched
class pysimavr.swig.simavr.avr_trace_data_t_old
```

```
    pc
    sp
class pysimavr.swig.simavr.avr_vcd_log_t
```

```
    signal
    value
    when
class pysimavr.swig.simavr.avr_vcd_signal_t
```

```
    alias
    irq
    name
    size
class pysimavr.swig.simavr.avr_vcd_t
```

```
    avr
    filename
    log
    logindex
    output
    period
    signal
    signal_count
    start
class pysimavr.swig.simavr.elf_firmware_t
```

```
    aref
    avcc
    bsssize
    codeline
    codesize
    command_register_addr
    console_register_addr
    datasize
```

```
    eeprom
    eesize
    flash
    flashbase
    flashsize
    frequency
    mmcu
    trace
    tracecount
    tracename
    traceperiod
    vcc

class pysimavr.swig.simavr.elf_firmware_t_trace

    addr
    mask
    name
```

6.2 high level interface

```
class pysimavr.ac.Ac(avr)

    getirq(pin)

class pysimavr.avr.Avr(firmware=None, mcu=None, f_cpu=None, avcc=5, vcc=5)

    arduino_targets = ['atmega48', 'atmega88', 'atmega168', 'atmega328p']
    avcc
    fpeek(addr)
    getirq(pin)
    goto_cycle(n)
    goto_time(tsec)
    load_firmware(firmware)
    move_time_marker(tsec_diff)
    pause()
    peek(addr)
    reset()
    run()
```

```
states = ['Limbo', 'Stopped', 'Running', 'Sleeping', 'StepStepDone']

step (n=1, sync=True)

terminate ()

time_passed ()

vcc

exception pysimavr.avr.UnkwownAvrError

pysimavr.connect.connect_irqs (irq_out, irq_in, bidirectional=False)

pysimavr.connect.connect_pins_by_rule (rule, device_map, vcd=None)
    rule example:
    B0 -> D4 -> vcd
    B1 <== D5 B2 ==> D6 #B3 <=> D7

class pysimavr.firmware.Firmware (filename=None)

    mcu
    read (filename)

class pysimavr.inverter.Inverter (avr)

    getirq (pin)
    out (i)

class pysimavr.lcd.Lcd (avr, size=(20, 2))

    get_char (x, y)
    getirq (pin)
    pinstate (pin)
    reset ()

class pysimavr.ledrow.LedRow (avr, size=8)

    getirq (pin)
    pinstate (i)
    reset_dirty (i)
        read and reset

class pysimavr.sgm7.Sgm7 (avr, size=4)

    digit_segments (digit_index)
    getirq (pin)
    pinindex (pin_name)
    pinstate (pin)
    reset_dirty (digit_index)
        read and reset
```

```
class pysimavr.vcdfile.VcdFile (avr, filename='gtkwave_output.vcd', period=10)

    add_signal (irq, name=None, bits=1)
    start ()
    stop ()
    terminate ()
```

DEVELOPMENT

7.1 Tools

1. `setuptools`
2. `Paver`
3. `nose`
4. `ghp-import`
5. `pyflakes`
6. `pychecker`
7. `paved fork`
8. `Sphinx`
9. `sphinxcontrib-programsscreenshot`
10. `sphinxcontrib-paverutils`
11. `autorun` from `sphinx-contrib` (there is no simple method, you have to download/unpack/setup)

7.2 Install on ubuntu

```
sudo apt-get install python-setuptools
sudo apt-get install python-paver
sudo apt-get install python-nose
sudo easy_install ghp-import
sudo apt-get install pyflakes
sudo apt-get install pychecker
sudo easy_install https://github.com/ponty/paved/zipball/master
sudo apt-get install scrot
sudo apt-get install xvfb
sudo apt-get install xserver-xephyr
sudo apt-get install python-imaging
sudo apt-get install python-sphinx
sudo easy_install sphinxcontrib-programsscreenshot
sudo easy_install sphinxcontrib-programoutput
sudo easy_install sphinxcontrib-paverutils
```

7.3 Tasks

[Paver](#) is used for task management, settings are saved in `pavement.py`. [Sphinx](#) is used to generate documentation.

print [paver](#) settings:

```
paver printoptions
```

clean generated files:

```
paver clean
```

generate documentation under *docs/_build/html*:

```
paver cog pdf html
```

upload documentation to [github](#):

```
paver ghpages
```

run unit tests:

```
paver nose
#or
nosetests --verbose
```

check python code:

```
paver pyflakes
paver pychecker
```

generate python distribution:

```
paver sdist
```

upload python distribution to [PyPI](#):

```
paver upload
```


INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

PYTHON MODULE INDEX

p

- `pysimavr.ac`, 17
- `pysimavr.avr`, 17
- `pysimavr.connect`, 18
- `pysimavr.firmware`, 18
- `pysimavr.inverter`, 18
- `pysimavr.lcd`, 18
- `pysimavr.ledrow`, 18
- `pysimavr.sgm7`, 18
- `pysimavr.swig.ac_input`, 11
- `pysimavr.swig.hd44780`, 11
- `pysimavr.swig.inverter`, 11
- `pysimavr.swig.ledrow`, 12
- `pysimavr.swig.sgm7`, 12
- `pysimavr.swig.simavr`, 12
- `pysimavr.vcdfile`, 18

INDEX

A

Ac (class in pysimavr.ac), 17
ac_input_t (class in pysimavr.swig.ac_input), 11
add_signal() (pysimavr.vcdfile.VcdFile method), 19
addr (pysimavr.swig.simavr.avr_symbol_t attribute), 13
addr (pysimavr.swig.simavr.elf_firmware_t_trace attribute), 17
alias (pysimavr.swig.simavr.avr_vcd_signal_t attribute), 16
arduino_targets (pysimavr.avr.Avr attribute), 17
aref (pysimavr.swig.simavr.avr_t attribute), 13
aref (pysimavr.swig.simavr.elf_firmware_t attribute), 16
avcc (pysimavr.avr.Avr attribute), 17
avcc (pysimavr.swig.simavr.avr_t attribute), 14
avcc (pysimavr.swig.simavr.elf_firmware_t attribute), 16
Avr (class in pysimavr.avr), 17
avr (pysimavr.swig.ac_input.ac_input_t attribute), 11
avr (pysimavr.swig.hd44780.hd44780_t attribute), 11
avr (pysimavr.swig.inverter.inverter_t attribute), 11
avr (pysimavr.swig.ledrow.ledrow_t attribute), 12
avr (pysimavr.swig.sgm7.sgm7_t attribute), 12
avr (pysimavr.swig.simavr.avr_io_t attribute), 12
avr (pysimavr.swig.simavr.avr_vcd_t attribute), 16
avr_io_t (class in pysimavr.swig.simavr), 12
avr_iopin_t (class in pysimavr.swig.simavr), 12
avr_ioport_getirq_t (class in pysimavr.swig.simavr), 12
avr_ioport_state_t (class in pysimavr.swig.simavr), 13
avr_ioport_t (class in pysimavr.swig.simavr), 13
avr_irq_pool_t (class in pysimavr.swig.simavr), 13
avr_irq_t (class in pysimavr.swig.simavr), 13
avr_kind_t (class in pysimavr.swig.simavr), 13
avr_symbol_t (class in pysimavr.swig.simavr), 13
avr_t (class in pysimavr.swig.simavr), 13
avr_t_cycle_timer (class in pysimavr.swig.simavr), 15
avr_t_io (class in pysimavr.swig.simavr), 15
avr_t_io_r (class in pysimavr.swig.simavr), 15
avr_t_io_shared_io (class in pysimavr.swig.simavr), 15
avr_t_io_shared_io_io (class in pysimavr.swig.simavr), 15
avr_t_io_w (class in pysimavr.swig.simavr), 15
avr_trace_data_t (class in pysimavr.swig.simavr), 15

avr_trace_data_t_old (class in pysimavr.swig.simavr), 16
avr_vcd_log_t (class in pysimavr.swig.simavr), 16
avr_vcd_signal_t (class in pysimavr.swig.simavr), 16
avr_vcd_t (class in pysimavr.swig.simavr), 16

B

bit (pysimavr.swig.simavr.avr_ioport_getirq_t attribute), 12
bsssize (pysimavr.swig.simavr.elf_firmware_t attribute), 16

C

c (pysimavr.swig.simavr.avr_t_io_r attribute), 15
c (pysimavr.swig.simavr.avr_t_io_shared_io_io attribute), 15
c (pysimavr.swig.simavr.avr_t_io_w attribute), 15
codeend (pysimavr.swig.simavr.avr_t attribute), 14
codeline (pysimavr.swig.simavr.avr_trace_data_t attribute), 15
codeline (pysimavr.swig.simavr.elf_firmware_t attribute), 16
codesize (pysimavr.swig.simavr.elf_firmware_t attribute), 16
command_register_addr (pysimavr.swig.simavr.elf_firmware_t attribute), 16
connect_irqs() (in module pysimavr.connect), 18
connect_pins_by_rule() (in module pysimavr.connect), 18
console_register_addr (pysimavr.swig.simavr.elf_firmware_t attribute), 16
count (pysimavr.swig.simavr.avr_irq_pool_t attribute), 13
cursor (pysimavr.swig.hd44780.hd44780_t attribute), 11
cycle (pysimavr.swig.simavr.avr_t attribute), 14
cycle_timer (pysimavr.swig.simavr.avr_t attribute), 14
cycle_timer_map (pysimavr.swig.simavr.avr_t attribute), 14
D
data (pysimavr.swig.simavr.avr_t attribute), 14
datapins (pysimavr.swig.hd44780.hd44780_t attribute), 11

datasize (pysimavr.swig.simavr.elf_firmware_t attribute), 16

ddr (pysimavr.swig.simavr.avr_ioport_state_t attribute), 13

dealloc (pysimavr.swig.simavr.avr_io_t attribute), 12

digit_count (pysimavr.swig.sgm7.sgm7_t attribute), 12

digit_pin (pysimavr.swig.sgm7.sgm7_t attribute), 12

digit_port (pysimavr.swig.sgm7.sgm7_t attribute), 12

digit_segments (pysimavr.swig.sgm7.sgm7_t attribute), 12

digit_segments() (pysimavr.sgm7.Sgm7 method), 18

digit_segments_changed (pysimavr.swig.sgm7.sgm7_t attribute), 12

E

e2end (pysimavr.swig.simavr.avr_t attribute), 14

EEPROM (pysimavr.swig.simavr.elf_firmware_t attribute), 16

eesize (pysimavr.swig.simavr.elf_firmware_t attribute), 17

eind (pysimavr.swig.simavr.avr_t attribute), 14

elf_firmware_t (class in pysimavr.swig.simavr), 16

elf_firmware_t_trace (class in pysimavr.swig.simavr), 17

F

filename (pysimavr.swig.simavr.avr_vcd_t attribute), 16

Firmware (class in pysimavr.firmware), 18

flags (pysimavr.swig.hd44780.hd44780_t attribute), 11

flags (pysimavr.swig.simavr.avr_irq_t attribute), 13

flash (pysimavr.swig.simavr.avr_t attribute), 14

flash (pysimavr.swig.simavr.elf_firmware_t attribute), 17

flashbase (pysimavr.swig.simavr.elf_firmware_t attribute), 17

flashend (pysimavr.swig.simavr.avr_t attribute), 14

flashsize (pysimavr.swig.simavr.elf_firmware_t attribute), 17

fpeek() (pysimavr.avr.Avr method), 17

frequency (pysimavr.swig.simavr.avr_t attribute), 14

frequency (pysimavr.swig.simavr.elf_firmware_t attribute), 17

fuse (pysimavr.swig.simavr.avr_t attribute), 14

G

gdb (pysimavr.swig.simavr.avr_t attribute), 14

gdb_port (pysimavr.swig.simavr.avr_t attribute), 14

get_char() (pysimavr.lcd.Lcd method), 18

getirq() (pysimavr.ac.Ac method), 17

getirq() (pysimavr.avr.Avr method), 17

getirq() (pysimavr.inverter.Inverter method), 18

getirq() (pysimavr.lcd.Lcd method), 18

getirq() (pysimavr.ledrow.LedRow method), 18

getirq() (pysimavr.sgm7.Sgm7 method), 18

goto_cycle() (pysimavr.avr.Avr method), 17

goto_time() (pysimavr.avr.Avr method), 17

H

h (pysimavr.swig.hd44780.hd44780_t attribute), 11

hd44780_t (class in pysimavr.swig.hd44780), 11

hook (pysimavr.swig.simavr.avr_irq_t attribute), 13

I

i_shadow (pysimavr.swig.simavr.avr_t attribute), 14

init (pysimavr.swig.simavr.avr_t attribute), 14

Inverter (class in pysimavr.inverter), 18

inverter_t (class in pysimavr.swig.inverter), 11

io (pysimavr.swig.simavr.avr_ioport_t attribute), 13

io (pysimavr.swig.simavr.avr_t attribute), 14

io (pysimavr.swig.simavr.avr_t_io_shared_io attribute), 15

io_port (pysimavr.swig.simavr.avr_t attribute), 14

io_shared_io (pysimavr.swig.simavr.avr_t attribute), 14

io_shared_io_count (pysimavr.swig.simavr.avr_t attribute), 14

ioctl (pysimavr.swig.simavr.avr_io_t attribute), 12

irq (pysimavr.swig.ac_input.ac_input_t attribute), 11

irq (pysimavr.swig.hd44780.hd44780_t attribute), 11

irq (pysimavr.swig.inverter.inverter_t attribute), 11

irq (pysimavr.swig.ledrow.ledrow_t attribute), 12

irq (pysimavr.swig.sgm7.sgm7_t attribute), 12

irq (pysimavr.swig.simavr.avr_io_t attribute), 12

irq (pysimavr.swig.simavr.avr_ioport_getirq_t attribute), 12

irq (pysimavr.swig.simavr.avr_irq_pool_t attribute), 13

irq (pysimavr.swig.simavr.avr_irq_t attribute), 13

irq (pysimavr.swig.simavr.avr_t_io attribute), 15

irq (pysimavr.swig.simavr.avr_vcd_signal_t attribute), 16

irq_count (pysimavr.swig.simavr.avr_io_t attribute), 12

irq_ioctl_get (pysimavr.swig.simavr.avr_io_t attribute), 12

irq_names (pysimavr.swig.simavr.avr_io_t attribute), 12

irq_pool (pysimavr.swig.simavr.avr_t attribute), 14

K

kind (pysimavr.swig.simavr.avr_io_t attribute), 12

L

Lcd (class in pysimavr.lcd), 18

LedRow (class in pysimavr.ledrow), 18

ledrow_t (class in pysimavr.swig.ledrow), 12

load_firmware() (pysimavr.avr.Avr method), 17

log (pysimavr.swig.simavr.avr_t attribute), 14

log (pysimavr.swig.simavr.avr_vcd_t attribute), 16

logindex (pysimavr.swig.simavr.avr_vcd_t attribute), 16

M

make (pysimavr.swig.simavr.avr_kind_t attribute), 13

mask (pysimavr.swig.simavr.elf_firmware_t_trace attribute), 17

mcu (pysimavr.firmware.Firmware attribute), 18
 mmcu (pysimavr.swig.simavr.avr_t attribute), 14
 mmcu (pysimavr.swig.simavr.elf_firmware_t attribute), 17
 move_time_marker() (pysimavr.avr.Avr method), 17

N

name (pysimavr.swig.simavr.avr_ioport_state_t attribute), 13
 name (pysimavr.swig.simavr.avr_ioport_t attribute), 13
 name (pysimavr.swig.simavr.avr_irq_t attribute), 13
 name (pysimavr.swig.simavr.avr_vcd_signal_t attribute), 16
 name (pysimavr.swig.simavr.elf_firmware_t_trace attribute), 17
 names (pysimavr.swig.simavr.avr_kind_t attribute), 13
 next (pysimavr.swig.simavr.avr_io_t attribute), 12
 next_cycle_timer (pysimavr.swig.simavr.avr_t attribute), 14

O

old (pysimavr.swig.simavr.avr_trace_data_t attribute), 15
 old_pci (pysimavr.swig.simavr.avr_trace_data_t attribute), 15
 out (pysimavr.swig.inverter.inverter_t attribute), 11
 out() (pysimavr.inverter.Inverter method), 18
 output (pysimavr.swig.simavr.avr_vcd_t attribute), 16

P

param (pysimavr.swig.simavr.avr_t_cycle_timer attribute), 15
 param (pysimavr.swig.simavr.avr_t_io_r attribute), 15
 param (pysimavr.swig.simavr.avr_t_io_shared_io attribute), 15
 param (pysimavr.swig.simavr.avr_t_io_w attribute), 15
 pause() (pysimavr.avr.Avr method), 17
 pc (pysimavr.swig.simavr.avr_t attribute), 14
 pc (pysimavr.swig.simavr.avr_trace_data_t_old attribute), 16
 pcint (pysimavr.swig.simavr.avr_ioport_t attribute), 13
 peek() (pysimavr.avr.Avr method), 17
 pending (pysimavr.swig.simavr.avr_t attribute), 14
 pending_wait (pysimavr.swig.simavr.avr_t attribute), 14
 period (pysimavr.swig.simavr.avr_vcd_t attribute), 16
 pin (pysimavr.swig.simavr.avr_iopin_t attribute), 12
 pin (pysimavr.swig.simavr.avr_ioport_state_t attribute), 13
 pinindex() (pysimavr.sgm7.Sgm7 method), 18
 pinstate (pysimavr.swig.hd44780.hd44780_t attribute), 11
 pinstate (pysimavr.swig.ledrow.ledrow_t attribute), 12
 pinstate (pysimavr.swig.sgm7.sgm7_t attribute), 12
 pinstate() (pysimavr.lcd.Lcd method), 18
 pinstate() (pysimavr.ledrow.LedRow method), 18
 pinstate() (pysimavr.sgm7.Sgm7 method), 18

pinstate_changed (pysimavr.swig.ledrow.ledrow_t attribute), 12
 pool (pysimavr.swig.simavr.avr_irq_t attribute), 13
 port (pysimavr.swig.simavr.avr_iopin_t attribute), 12
 port (pysimavr.swig.simavr.avr_ioport_state_t attribute), 13
 pysimavr.ac (module), 17
 pysimavr.avr (module), 17
 pysimavr.connect (module), 18
 pysimavr.firmware (module), 18
 pysimavr.inverter (module), 18
 pysimavr.lcd (module), 18
 pysimavr.ledrow (module), 18
 pysimavr.sgm7 (module), 18
 pysimavr.swig.ac_input (module), 11
 pysimavr.swig.hd44780 (module), 11
 pysimavr.swig.inverter (module), 11
 pysimavr.swig.ledrow (module), 12
 pysimavr.swig.sgm7 (module), 12
 pysimavr.swig.simavr (module), 12
 pysimavr.vcdfile (module), 18

R

r (pysimavr.swig.simavr.avr_t_io attribute), 15
 r_ddr (pysimavr.swig.simavr.avr_ioport_t attribute), 13
 r_pcint (pysimavr.swig.simavr.avr_ioport_t attribute), 13
 r_pin (pysimavr.swig.simavr.avr_ioport_t attribute), 13
 r_port (pysimavr.swig.simavr.avr_ioport_t attribute), 13
 ramend (pysimavr.swig.simavr.avr_t attribute), 14
 rampz (pysimavr.swig.simavr.avr_t attribute), 14
 read() (pysimavr.firmware.Firmware method), 18
 readpins (pysimavr.swig.hd44780.hd44780_t attribute), 11
 reset (pysimavr.swig.simavr.avr_io_t attribute), 12
 reset (pysimavr.swig.simavr.avr_t attribute), 14
 reset() (pysimavr.avr.Avr method), 17
 reset() (pysimavr.lcd.Lcd method), 18
 reset_dirty() (pysimavr.ledrow.LedRow method), 18
 reset_dirty() (pysimavr.sgm7.Sgm7 method), 18
 run (pysimavr.swig.simavr.avr_t attribute), 14
 run() (pysimavr.avr.Avr method), 17

S

segment_pin (pysimavr.swig.sgm7.sgm7_t attribute), 12
 segment_port (pysimavr.swig.sgm7.sgm7_t attribute), 12
 Sgm7 (class in pysimavr.sgm7), 18
 sgm7_t (class in pysimavr.swig.sgm7), 12
 signal (pysimavr.swig.simavr.avr_vcd_log_t attribute), 16
 signal (pysimavr.swig.simavr.avr_vcd_t attribute), 16
 signal_count (pysimavr.swig.simavr.avr_vcd_t attribute), 16
 signature (pysimavr.swig.simavr.avr_t attribute), 14
 size (pysimavr.swig.simavr.avr_vcd_signal_t attribute), 16

[sleep \(pysimavr.swig.simavr.avr_t attribute\), 14](#)
[sp \(pysimavr.swig.simavr.avr_trace_data_t_old attribute\), 16](#)
[special_deinit \(pysimavr.swig.simavr.avr_t attribute\), 14](#)
[special_init \(pysimavr.swig.simavr.avr_t attribute\), 14](#)
[sreg \(pysimavr.swig.simavr.avr_t attribute\), 14](#)
[start \(pysimavr.swig.simavr.avr_vcd_t attribute\), 16](#)
[start\(\) \(pysimavr.vcdfile.VcdFile method\), 19](#)
[state \(pysimavr.swig.simavr.avr_t attribute\), 15](#)
[states \(pysimavr.avr.Avr attribute\), 17](#)
[step\(\) \(pysimavr.avr.Avr method\), 18](#)
[stop\(\) \(pysimavr.vcdfile.VcdFile method\), 19](#)
[symbol \(pysimavr.swig.simavr.avr_symbol_t attribute\), 13](#)

T

[terminate\(\) \(pysimavr.avr.Avr method\), 18](#)
[terminate\(\) \(pysimavr.vcdfile.VcdFile method\), 19](#)
[time_passed\(\) \(pysimavr.avr.Avr method\), 18](#)
[timer \(pysimavr.swig.simavr.avr_t_cycle_timer attribute\), 15](#)
[touched \(pysimavr.swig.simavr.avr_trace_data_t attribute\), 16](#)
[trace \(pysimavr.swig.simavr.avr_t attribute\), 15](#)
[trace \(pysimavr.swig.simavr.elf_firmware_t attribute\), 17](#)
[trace_data \(pysimavr.swig.simavr.avr_t attribute\), 15](#)
[tracecount \(pysimavr.swig.simavr.elf_firmware_t attribute\), 17](#)
[tracename \(pysimavr.swig.simavr.elf_firmware_t attribute\), 17](#)
[traceperiod \(pysimavr.swig.simavr.elf_firmware_t attribute\), 17](#)

U

[UnkownAvrError, 18](#)
[used \(pysimavr.swig.simavr.avr_t_io_shared_io attribute\), 15](#)

V

[value \(pysimavr.swig.ac_input.ac_input_t attribute\), 11](#)
[value \(pysimavr.swig.simavr.avr_irq_t attribute\), 13](#)
[value \(pysimavr.swig.simavr.avr_vcd_log_t attribute\), 16](#)
[vcc \(pysimavr.avr.Avr attribute\), 18](#)
[vcc \(pysimavr.swig.simavr.avr_t attribute\), 15](#)
[vcc \(pysimavr.swig.simavr.elf_firmware_t attribute\), 17](#)
[vcd \(pysimavr.swig.simavr.avr_t attribute\), 15](#)
[VcdFile \(class in pysimavr.vcdfile\), 18](#)
[vector \(pysimavr.swig.simavr.avr_t attribute\), 15](#)
[vector_size \(pysimavr.swig.simavr.avr_t attribute\), 15](#)
[vram \(pysimavr.swig.hd44780.hd44780_t attribute\), 11](#)

W

[w \(pysimavr.swig.hd44780.hd44780_t attribute\), 11](#)
[w \(pysimavr.swig.simavr.avr_t_io attribute\), 15](#)